NASA-CR-195230

THE STATE UNIVERSITY OF NEW JERSEY
RUTGERS

NASA Ames Research Center: Grant NAG 2-829

Feature Isolation and Quantification of Evolving Datasets

Overview of Research, as of February 1994

General Abstract

Identifying and isolating features is an important part of visualization and a crucial step for the analysis and understanding of large time-dependent data sets (either from observation or simulation). In this proposal, we address these concerns, namely the investigation and implementation of basic 2D and 3D feature based methods to enhance current visualization techniques and provide the building blocks for automatic feature recognition, tracking, and correlation. These methods incorporate ideas from scientific visualization, computer vision, image processing, and mathematical morphology. Our focus is in the area of fluid dynamics, and we show the applicability of these methods to the quantification and tracking of three-dimensional vortex and turbulence bursts.

Past Accomplishments

The following is a short description of the work which has been previously completed. (A detailed description is contained in the accompanying manuscript, *Feature Tracking and Visualization*).

The feature extraction algorithm, which performs a 3D segmentation on the scalar and vector fields, has been implemented as a stand-alone program which can be integrated into any existing visualization package. The output is a set of regions, with computed characteristics, such as volume, mass, moments, average vorticity, bounding surface, etc. The feature tracking routine automatically reads in the segmented regions and characteristics for a series of time steps and correlates the regions based upon the characteristics (*correspondence*). The output from this program is a graph of the region histories, or a visualization of the histories using color-coded bounding surfaces of region evolutions. The feature tracker has been demonstrated on all 816 time-steps from a CFD simulation. Besides being essential for quantification, the feature extraction greatly enhances visualization. While a few evolving regions can be followed visually, with many amorphous structures it is difficult to gain insight and distinguish between different interacting regions. Objects can be color-coded so that their evolutions can be correlated, or just rendered alone for better comprehension.

Recent Accomplishments

The following is a short description of the work which has been completed as of February 1994:

The feature extraction algorithm has been implemented on a parallel supercomputer (CM5) to facilitate the handling of massive datasets. Many simulations involve numerous time-steps and it is difficult to routinely store this information on a local workstation for later visualization and analysis. By performing some of the feature extraction on the supercomputer where the data resides, the massive data handling problem is alleviated.

A skeletal program to find the medial axis or specialized skeleton ("vortex core") has been implemented. The program tracks a vortex core using a "predictor-corrector" method. The algorithm begins by following the vector direction from a local maxima. At each grid intersection, the magnitude of the same or a different variable at the intersection and neighboring grid locations is checked, and the largest (or smallest depending upon the variable being searched) is selected as the next point in the core. The process repeats until the core is isolated. (A paper describing the algorithm is being prepared and will be completed soon.)

Current Efforts

Our current efforts include improving on the existing algorithm and researching and implementing new ones. We are currently optimizing the skeletal algorithm for different datasets and variables. All of the algorithms implemented so far (the feature extraction and tracking) have been for regular gridded datasets. We are enhancing them to handle non-regular gridded data. These routines will also be implemented on a parallel supercomputer. In addition, we are experimenting with a distributed-workstation environment (PVM) for the implementation.

Feature Tracking and Visualization

R. Samtaney[†] & D. Silver [*] & N. Zabusky[†] & J. Cao[*]
c/o Laboratory for Visiometrics and Modeling
CAIP Center
Rutgers University
PO BOX 1390
Piscataway, NJ 08855-1390
(908)932- 5546 (Silver), x5869 (Zabusky)
FAX: 908-932-4475
samtaney/silver/nzabusky/jcao @vizlab.rutgers.edu

Abstract

An essential part of visualizing massive time-dependent data sets is to identify, quantify and **track** important regions and structures (objects of interest). This is true for almost all disciplines since the crux of understanding the original simulation, experiment or observation is the study of the evolution of the "objects" present. Some well known examples include tracking the progression of a storm, the motion and change of the "ozone hole", or the movement of vortices shed by the meandering Gulf stream. In this paper, we present techniques to track two and three dimensional evolving objects in time dependent simulations. The results can enhance visualization by highlighting areas of activity and following regions of interesting evolution. The techniques described in the paper are applied to data from ongoing research in computational fluid dynamics (CFD), however, the tracking procedures are general and appropriate for many other disciplines.

Keywords: visualization, feature extraction, feature tracking, computer vision, CFD.

Black and White version (original in color)
Submitted August 1993, Revised March 1994

[†] Dept. of Mechanical and Aerospace Engineering, Rutgers University.
[*] Dept. of Electrical and Computer Engineering, Rutgers University

1 Introduction and Motivation

The aim of the visualization of massive scientific data sets is to devise algorithms and methods which transform numerical data into pictures and other graphic representations, thereby facilitating comprehension and interpretation. In these pictures, regions of activity are observed. In many domains, it is the analysis of these regions which prompted and motivated the scientific investigation - a laboratory experiment, numerical simulation, or remote observation.

If a particular region is of interest, the scientist attempts to identify and quantify it: what is it, what is its cause, how does it evolve, how long does it persist, etc. The aim in the different disciplines is to study the evolution and essential dynamics of these objects and describe them for modified time periods, thus obtaining a partial solution or reduced-and-simpler model of the original problem. For example, one tracks the progression of a storm for weather prediction, the change in the ozone "hole" for knowledge about the greenhouse effect, or the movement of air over an aircraft or automobile for better design.

Unfortunately, this is a daunting task because the data generated is overwhelming. For example, a typical 3D numerical unsteady simulation in Computational Fluid Dynamics (CFD) may involve hundreds of time steps on the order of 256^3 or greater. Most scientists cannot routinely store this information locally or have it accessible interactively for visualization and analysis. Performing the visualization during the computation may reduce the storage space and post-processing time. However, there still may be too much information to absorb since many of these datasets are very "busy". Furthermore, most standard visualization procedures concentrate upon rendering a dataset and not quantifying the numerous observed regions. Because the scientist is primarily interested in higher level phenomena, a workable solution to the data problem is to focus on just those "features" - i.e. automatically extracting and tracking them. This both reduces the amount of data and provides a crucial first step to help understand their evolution. In addition, extracted regions can be used for database management to store and retrieve datasets based upon content. This involves extraction, tracking and classification. For example, in Figure 1 five out of 816 time-dependent datasets are shown, and it is apparent that objects (features) are evolving. This can be seen in the graph of the evolution of the objects in Figure 7. A color-coded and tracked sequence resulting from this graph is shown in Figure 8; and in Figure 9, one particular region was isolated for further study. The examples are explained in Section 4.

Related Work

Feature based tools are typical in 2D image processing and computer vision [1, 2]. There has been some work in feature extraction in 3D (see for example [3, 4, 5] or [6] for a more complete review). The majority has been related to medical imaging. In most of these methods, seed-expansion is used to isolate one distinct region. Some of these techniques will be described later.

Tracking objects in a series of two-dimensional images is widely dealt with in computer vision [1], i.e. *motion tracking* and *optical flow*. The major issue is to find a particular feature in a series of consecutive frames. The process of matching an object in one frame to one in another is called the *correspondence problem*. The objects are generally matched using a range of attributes such as pixel values, edges, moments, geometry, etc. While many techniques are applicable [7], there are some differences between the two fields primarily because scientific "objects" evolve and interact and the problem of projections from three- to two-dimensions (to create a picture) does not exist. Some scientific domains such as oceanography and meteorology have incorporated tracking to process the remote sensing observations which are continually being generated. For example, in [8], cloud tracking is performed by calculating attributes of the clouds and searching for matches in the next dataset. In fluid dynamics, tracking has been performed on vortex tubes by first determining the core of the tube in one dataset. A window about the position of the core was then used to located the core in the next dataset [9].

Time tracking is analogous to space tracking, i.e. given a set of 2D contour-slices representing a continuous 3D domain, determine the correspondence between the surfaces from one slice to the next. The characterization of possible scenarios and topologies is similar (see [10] for one example), although generally only edges are matched instead of entire regions.

In this paper, we describe *basic* algorithms to extract coherent amorphous regions (*features or objects*) from two and three dimensional scalar and vector fields and then to track them in a series of consecutive time steps. A combination of techniques from computer vision, image processing, computer graphics, and computational geometry are used, and these are described in Sections 2 and 3. In Section 4, the techniques are applied to datasets from Computational Fluid Dynamics, however, the results can be generalized to other disciplines with continuous time-dependent scalar (and vector) fields.

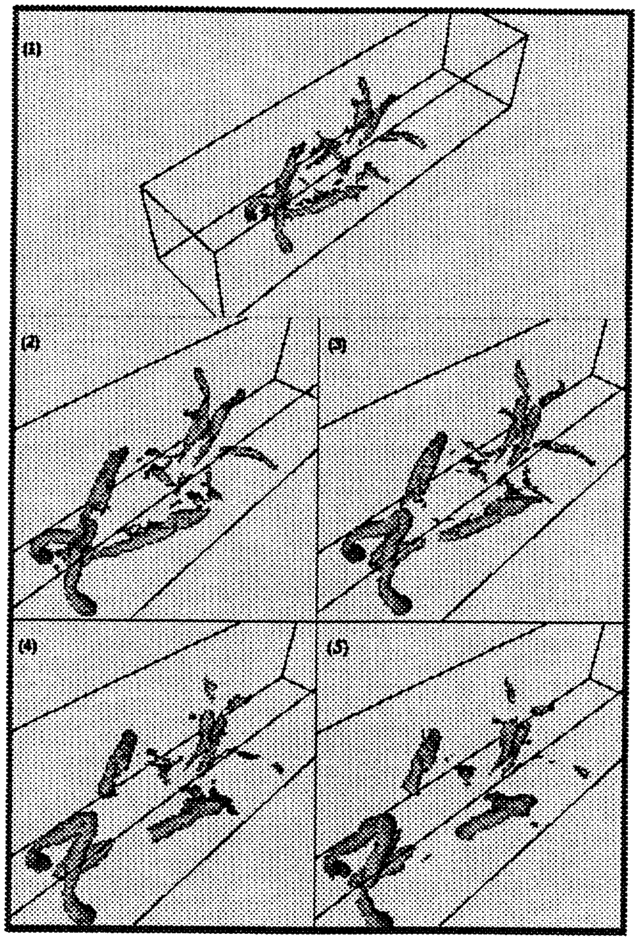

FIGURE 1. *Vorticity isosurface for 3D shock-interface interaction. Time = (1) 200, (2) 220, (3) 240, (4) 260, (5) 280*

2 Defining Features

Each domain has its own set of interesting *objects or features.* These are usually defined as a *region* of the dataset which satisfies certain constraints, for example, an area of low pressure may define an oncoming storm. Standard visualization programs highlight iso-valued clusters since the eye is naturally drawn to colored coherent areas. This is the simplest definition of a feature, and one that is common to many areas of scientific research. In this definition, objects consist of a set of *neighboring* interior points above or below a certain threshold value and their boundaries. While many other types of features are of interest (for example, vector field lines, critical points [11], etc.) in what follows we concentrate on thresholded clusters.

Connected thresholded regions can be extracted using a 3D segmentation or region-growing algorithm [6]. If the region is to contain "high" values, local maxima may be used as *seed* nodes. The 3D neighbors of the seeds are then recursively tested for inclusion in the regions. When a node is hit which is not above the chosen threshold value, the region stops growing. The dataset will then be partitioned into "objects" and background. The set of nodes which comprise the object should be stored in a data structure for efficient manipulation. For regular gridded datasets an octree [12] is effective. Seed nodes can be interactively chosen by the user, or automatically generated by stepping through different threshold values from the maximum to the minimum. The shape and size of the region can also be controlled by multidimensional thresholding, topological parameters known about the domain, or using a gradient filter to define the "edges".

In Figure 1(1) an isosurface of a dataset is shown. This is a widely used visualization method. The dataset is a scalar field, derived from a vorticity vector field, with dimensions 256×64^2. (NOTE: the dataset is from a sequence of 816 time steps. Five from this series are displayed in the figure.) Different regions are evident, but are not clear and readily accessible for quantification. Figure 8(1) also contains the isosurfaces, however, in this case, the large connected regions were isolated and stored. Now, attributes of each region (see next section) can be calculated. Since they are all separate regions, the boundaries are distinct and can be colored differently.

2.1 Object Attributes

Attributes are useful for quantifying the extracted regions (i.e. a set of nodes satisfying certain constraints), and for tracking. Some common attributes include: [1]

- *Mass.* The mass is the weighted sum of all the nodes contained within the extracted region,

$$W = \int_{\Omega} w(\mathbf{x})\ d\Omega \tag{1}$$

  where $\mathbf{x} = (x_1, x_2, \cdots, x_d)$ for the dimension of the field, $w(\mathbf{x})$ is the scalar value at the node $\mathbf{x}$, and $\Omega = \{\mathbf{x} | \omega(\mathbf{x}) > T_\omega\}$ for a particular threshold value T_ω.

- *Centroid.* The weighted average of all the points in the isolated region is the centroid (the centroid may not be within the boundaries of the object), namely:

$$\overline{x_k} = W^{-1} \int_{\Omega} w(\mathbf{x})\ x_k\ d\Omega \quad (k = 1, 2, ..., d) \tag{2}$$

- *Maximum.* An extracted region may have several local maxima. These can be detected with the seed growing algorithm.

- *Volume.* The number of nodes contained within the region is an approximation to the area or volume.

- *Moment.* The moments can be used to characterize the shape and orientation of the region. The second moments define an ellipse in 2D or an ellipsoid in 3D:

$$I_{ij} = W^{-1} \int_{\Omega} w(\mathbf{x})\ (x_i - \overline{x_i})(x_j - \overline{x_j})\ d\Omega \tag{3}$$

  I_{ij} is a $d \times d$ matrix where $i = 1, 2, ..., d$ and $j = 1, 2, ..., d$.

- *Domain Specific Variables.* Each domain or dataset will have different computable characteristics depending upon what other variables are available, examples include average velocity or vorticity, circulation, temperature, pressure, etc.

[1]the following definitions all assume regular datasets

- *Bounding surface.* The bounding surface of the extracted region is similar to the *isosurface* if the region was extracted by simple thresholding (except that each separate region is characterized by its own set of polygons). Bounding nodes can be tagged during region growing to speed up the computation.

While second-order moments are a good approximation to *blob-like* regions, they do not accurately capture tube or sheet like structures. Higher order moments may provide a better approximation as well as providing an error measure for the second moments. Skeletons (spines) are also useful to abstract regions and characterize vortex cores [9].

3 Tracking Features

Tracking can be performed in either a *pre-processing* or *post-processing* mode. In *post-processing*, regions are first extracted from all the datasets, and then correlated. In the *pre-processing*, the regions extracted from one dataset are used to search for the regions in the next dataset (see [9] for an example). In what follows, we concentrate on the post-processing method. Since objects may move or change significantly, some simple qualifications must be stated to limit the range and number of possible scenarios. The most basic assumption is that the time between successive datasets, $\Delta t = t_{i+1} - t_i$, is small.

3.1 Interactions

During any experiment objects evolve. The evolutionary events can be characterized as follows:

1. Continuation - an object continues from time t_i to t_{i+1} with possible rotation or translation, its size may remain the same, intensify (become larger - grow), or weaken (begin to dissipate);

2. Creation - new objects appear above the threshold;

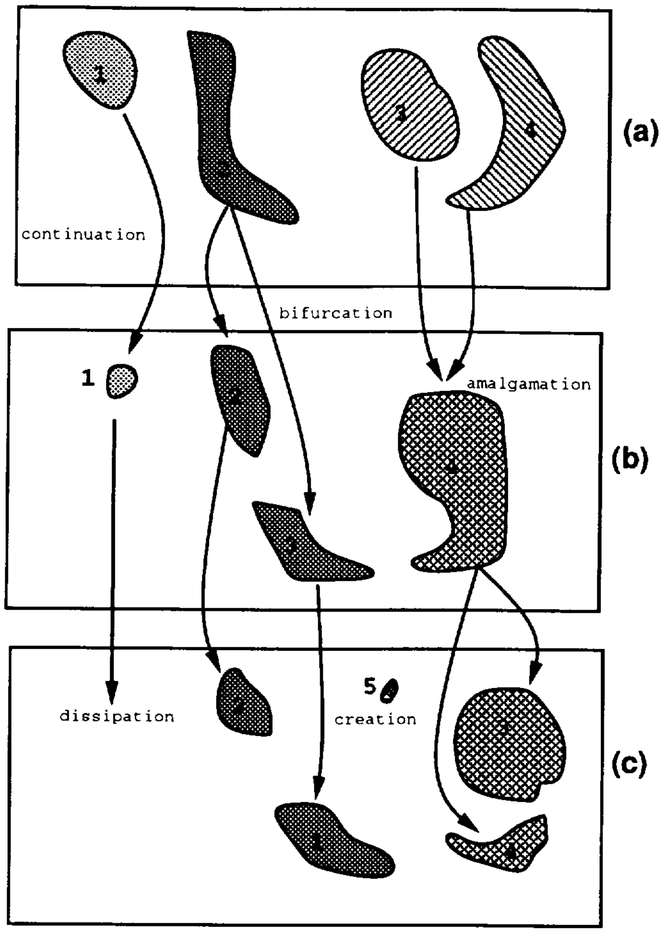

FIGURE 2. *Tracking interactions: continuation, creation, dissipation, bifurcation and amalgamation.*

3. Dissipation - an object disappears;

4. Bifurcation - an objects separates into two or more substructures;

5. Amalgamation - two or more objects merge.

These actions are illustrated in Figure 2.

3.1.1 Continuation

A feature which remains the same from one time step to another is said to continue. To determine whether a particular object in one dataset "continues" as an object in a subsequent dataset, the *correspondence criteria* need to be fully defined. In what follows, we present a basic notion of correspondence for our scientific domain. This is a general notion, and is applicable in many other domains as well.

We define correspondence based upon the intuitive notion of the scientist who has studied the evolution of these amorphous regions in the past. For example, in Figure 2 one will immediately notice that O_1^a, *object 1* at time $t = a$, [2] continues as O_1^b, and $O_3^b \rightarrow O_1^c$.

Definition 1 *Object O_B^{i+1} corresponds to O_A^i if*
$$\{O_B^{i+1} | overlap(O_A^i, O_B^{i+1}) > overlap(O_A^i, O_G^{i+1}) \ for \ all \ O_{G \neq B}^{i+1} \in \mathbf{O_N^{i+1}}\}$$

i.e. the nodes of object O_B^{i+1} have maximum overlap (location and value) with those of object O_A^i. A second condition is sometimes necessary: $overlap(O_A^i, O_B^{i+1}) > T_{over}$, where T_{over} is a threshold on the size of the intersection.

The condition above requires storing the objects in their entirety. Since the set of nodes which define the objects may consume a great deal of storage (even if only two time steps are processed at once), it is sometimes more convenient to save only the computed attributes, such as distance and volume, and use those to determine correspondence. Either the *closest* is chosen, i.e. attribute(O_B^{i+1}) which is most similar to attribute(O_A^i) from all the other objects in $\mathbf{O_N^i}$; or the relative difference between the objects is below a set threshold or threshold percentage. For moving regions where the velocity information is available, objects can be rotated or translated before comparing.

3.1.2 Bifurcation and Amalgamation

If an object splits into two or more substructures (in the next time step), that object is said to bifurcate. Similarly, amalgamation occurs if two or more objects merge.

Definition 2 *If a group of N objects ($N > 1$), $\mathbf{S_N^{i+1}}$* added *together are equivalent to O_A^i, then O_A^i has bifurcated at time $i+1$ into the regions of $\mathbf{S_N^{i+1}}$.*

The equivalence criteria is defined above. The mass, volume, and circulation can be added and compared to the original object. The weighted centroids of the bifurcated regions

[2]Notation: the bold $\mathbf{O_N^i}$ refers to a set, $\mathbf{O}$ of N objects extracted at time $t = i$, whereas, O_A^i refers to a particular object, labeled A, at time $t = i$.

can be compared to the original object. (If O_A^i has bifurcated into the set $\mathbf{S}_{\mathrm{N}}^{\mathbf{i+1}}$, then $\sum attribute(S_{a\in N}^{i+1}) - attribute(O_A^i) < T_{attribute}$.) Amalgamation is the inverse property, namely:

Definition 3 *If a group of objects $\mathbf{S}_{\mathrm{N}}^{\mathbf{i}}$ added together are equivalent to O_A^{i+1}, then the regions of $\mathbf{S}_{\mathrm{N}}^{\mathbf{i}}$ have merged into object O_A^{i+1}. This is called amalgamation.*

In Figure 2, O_2^a has bifurcated into the two objects O_2^b and O_3^b. Similarly, O_3^a and O_4^a merge to become O_4^b.

3.1.3 Creation and Dissipation

Creation occurs when the an object in one time step can not be correlated with any object in a previous time. An object from one time step **dissipates** when there is no object in the subsequent time step which can be matched to it.

Definition 4 *If an object O_A^{i+1}, is not a continuation of an object at time t_i, and has not been classified as part of a merge or bifurcation, then O_A^{i+1} is a new object (creation).*

Definition 5 *If an object O_A^i, cannot be correlated to any object at t_{i+1}, and has not been classified as part of an amalgamation or bifurcation, then O_A^i has dissipated.*

In Figure 2, O_1^b dissipates in the next time step, and O_5^c appears for the first time. Dissipation (creation) occurs when regions fall below (above) the chosen threshold value.

3.2 Tracking

The algorithm described below performs a basic matching and relies upon centroid, mass, volume and circulation (in 2D). Tolerances are used to determine the "goodness" of the match. Similar algorithms are listed in [8, 1].

1. Extract the objects from each dataset, numbering each object and maintaining a list of attributes.

2. Starting with the first and second dataset, compare every dataset with a subsequent one.

3. For each object (centroid) in dataset i, calculate which centroids from dataset, $i+1$, is closest to it and test whether the volume, mass, etc are within the prescribed tolerances ($T_{attribute}$). If a match is found, tag those objects and remove them from the list.

4. After all the objects which continue have been removed, test combinations of two, three, etc from t_{i+1} for bifurcation, and amalgamation. Bifurcation and Amalgamation are determined based upon the difference between the average weighted centroids, total volume and total mass of the combinations with the original.

To minimize the matching process, only large regions can be tracked. Since the number of combinations for amalgamation and bifurcation may still be larged, testing can be limited to close regions (defined by a tolerance) or using neighborhoods. For example, in Figure 2a, object 4 is unlikely to merge with object 1 in the next time step. For objects at time t_i and t_{i+1}, the closest object or nearest neighbor to O_A^i is that which minimizes $dist(O_A^i, O_B^{i+1})$. The next level of neighborhood is defined as those objects in t_{i+1} which are closer to O_A^i in a particular direction than any other objects. This can be determined by constructing a Voronoi diagram of the objects in t_{i+1} with O_A^i using the centroids, (or approximately using windows and distance measures), and extracting the neighboring cells of O_A^i. The next level of neighborhood, is the neighbors of neighbors which can be found recursively. The process continues until all the objects are included.

3.3 Visualizing Object Histories

After the tracking procedure, the history of each object is known. Different representations can be used to display this information. For example, the histories of objects in Figure 2 can be characterized as follows:

a,b: $1(a) \rightarrow 1(b)$, $2(a) \rightarrow 2(b) + 3(b)$, $3(a) + 4(a) \rightarrow 4(b)$;

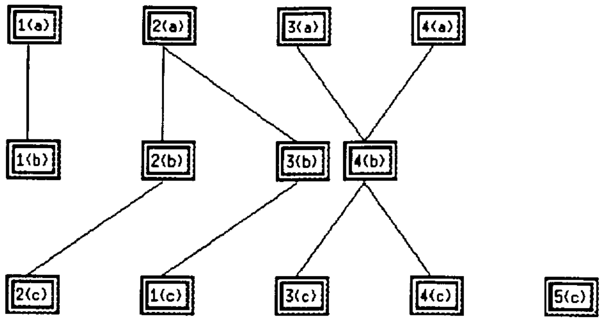

FIGURE 3. *DAG History of the evolution of observed regions.*

b,c: $2(b) \rightarrow 2(c)$, $3(b) \rightarrow 1(c)$, $4(b) \rightarrow 3(c) + 4(c)$;

In Figure 3, the information is represented as a directed acyclic graph (DAG). A legend of the object names and their attributes is needed for complete understanding of these representations (the file used by the tracking program stores the object legends.) Alternately, for 2D, a plot of the $x - centroid$ position with respect to time can also highlight evolution (and position) as seen in Figure 6.

Objects can also be colored by their histories either by assigning their descendents the same color or by shading descendents as a percentage of their parents color (e.g. by volume). Both the DAG and X-centroid plot can be color coded with the objects.

3.4 Implementation

The current implementation of the program is divided into two parts, a feature extractor and a feature tracker. The feature extractor performs thresholding and segmentation (it works with the simulation or as a module in AVS). A set of rules can be supplied to further enhance the segmentation [6]. The output from this program is a file specifying each object and the attributes describing the object. The bounding surface can be used to view the isosurface.

The feature tracker reads in the set of objects and attributes from each data set and then performs the correspondence in two or three dimensions. Distance is used as the primary matching parameter, with the other attributes as secondary checks. The output is a either a text file containing the history, a DAG, or a set of color coded objects.

4 Examples

In this section, we demonstrate feature extraction and tracking on two examples from ongoing research in CFD. Both are simulations of a *Shock-Interface* interaction. One is a 2D simulation and the other is 3D.

4.1 The Shock-Interface Problem

One of the fundamental interactions in compressible hydrodynamics is between a shock wave and a density inhomogeneity. The physical situation, shown in Figure 4, may be characterized by a shock wave propagating through a fluid of density, ρ_1, striking a contact interface and passing into a region of density, ρ_2. The governing equations are the compressible Euler equations. The physical processes can be divided into two phases: a rapid vorticity [3] deposition phase; and a vorticity evolution phase during which the interface is characterized by the presence of coherent vortex structures (CVS) which are the "features" or "objects" of interest to the fluid dynamicist. Quantification of properties (such as circulation, area, centroid) of the CVS and their interactions with each other are essential for a physical understanding and development of reduced models of the ensuing turbulent mixing phase. (For further details about this problem see [13].)

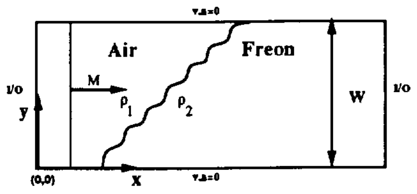

FIGURE 4. *Schematic of the Shock-Interface Problem*

4.2 2D Example

In the 2D shock-interface simulation, the interface is Air-Freon inclined at 60° to the vertical with an incident shock Mach number, $M = 1.5$. The datasets are 800×160 with uniform

[3]Vorticity is defined as $\omega = \nabla \times u$, where u is the velocity field.

unity grid spacing. The quantity being studied is the vorticity field (vorticity is a vector quantity, however, in the following two examples the magnitude of the vorticity field is being used.) In Figure 5, four of the 3200 images are shown at times: $t_a = 160$, $t_b = 480$, $t_c = 800$, and $t_d = 1280$. As the shock traverses the interface, vorticity is deposited on the interface, this can be seen as the dark region in t_a. Four CVS are then formed, which later amalgamate. The amalgamation of CVS (1) and (2) is observed in t_d. To summarize, the entire physical process is characterized by the generation of a vortex layer followed by splitting, filamentation, and finally amalgamation. The four dominant CVS are tracked with the threshold value, $T_\omega = 0.0225$. The filamentary structures which lie above the threshold are not tracked in order to focus on the cores of the CVS. The history of the objects can be represented by plotting the x and y centroid of each CVS as a function of time (see Figure 6) or as a DAG. The choice of the threshold value is, in general, domain dependent. In this particular case, the threshold is chosen based on the theoretical total vorticity on the interface [13]. One of the important quantities in the domain is the total negative circulation (Γ_-), i.e. the sum of all the negative vorticity. It was observed that more than 75% (50%) of Γ_- was concentrated within the tracked CVS for $t < 500$ ($t > 500$).

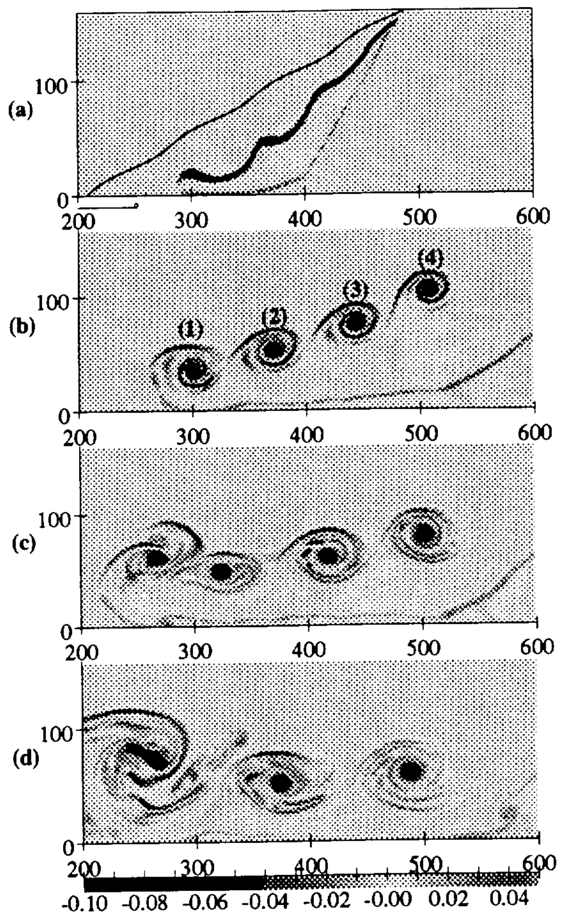

FIGURE 5. *2D Shock-Interface Air-Freon, vorticity plot at time =(a)160, (b)480, (c)800, (d)1280*

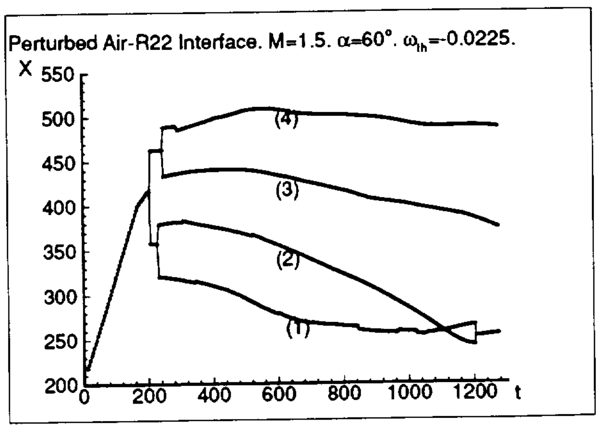

FIGURE 6. *Feature Tracking, 2D Shock Interface: plot of the CVS centroid (X value) vs. time.*

4.3 3D Example

A three dimensional $M = 2.0$ shock interaction with an Air-Freon interface is shown in Figure 1 and in 8. The planar interface is inclined at 45^o to the plane of the shock, and the datasets are 256×64^2 of vorticity magnitude.

Due to a physical phenomenon called *vortex stretching* which is absent in 2D, the topologies of the CVS are more complex in this case. In Figure 1, isosurfaces ($T_\omega = 0.15$) of five of the 816 datasets from the simulation are presented at $t_1 = 200$, $t_2 = 220$, $t_3 = 240$, $t_4 = 260$, $t_5 = 280$. Only the large-scale regions are of interest here, so "small" regions were disregarded. The extracted regions were tracked in consecutive datasets and the overall history of the five data sets is shown in the DAG in Figure 7 and the color coded history is shown in Figure 8. At t_1, seven coherent vortices (colored differently) are observed in the flow field. The most complex vortex structure, labeled 0, is in the shape of a half vortex ring with two tails. The evolution of this structure is important in fluid dynamics. At t_2, CVS O_0^1 has split into three part, the two tails, O_1^2 and O_2^2, and the ring portion, O_0^2. Note that the three objects in the middle of the domain have dissipated. As time progresses, both O_1^2 and O_2^2 bifurcate once more, and the tip portion amalgamates at t_3. A movie of all the datasets was made with each region color coded. In Figure 9, O_1^2 is tracked and rendered by itself.

5 Conclusion

The purpose of this work was to demonstrate how some basic segmentation and tracking algorithms could enhance visualization and analysis of large time-dependent data sequences. We are currently improving the algorithm and testing other more complex methods to determine the best techniques for large 3D scientific datasets. Domain dependent identification parameters may be appropriate in certain instances. Centroids can be misleading, as in the case of a torus and an object in the center, where both will have the same centroid. The correspondence problem could be improved by using *template matching*, voxel to voxel based comparisons if all the information can be stored, or *optimization techniques* (paramater space matching) to find the best match over a number of parameters.

When Δt is small, less discontinuities in the history plots resulted. As Δt increases, objects move farther and their volume changes more rapidly making it harder to correlate them. While it is desirable to have small time steps, it is not always available (especially if the feature extractor is not implemented with the simulation). In this case, the various tolerances, $T_{attribute}$, must increase. A common error which occurs when the threshold is too low is that regions will be tagged as continuing when they had a actually bifurcated into two regions, one large (almost the size of the original) and one small region. The small region is then regarded as a "new" object (creation). In many simulations, creation is usually the result of bifurcation. Therefore, if an object is tagged for creation, it can be combined to neighboring objects to see if the error is minimized when added. We are currently performing more experiments to determine the sensitivity of the algorithm to Δt.

While the DAG representation and other plots are useful, they must be correlated to the extracted object information. An interactive interface would be helpful to highlight the objects visually. For example, if a node on the DAG is chosen, the object (and the history) corresponding to that node will be rendered. Defining features is also an important area of study. Each domain has its own set of *interesting* features with parameters to define the feature. Once the features are defined they can be classified and stored for later use. One can envision a sophisticated database for scientific applications where events found in one simulation can be searched for in others and then automatically rendered.

The ultimate goal of visualization is to aid in the understanding and analysis of data. With the advent of faster parallel computers, more sophisticated sensing devices, and higher bandwidth communication channels, information is being produced in ever greater amounts. This information must be presented to the scientist in a form suitable for cogent assimilation. /samepage There is an urgent need for better tools to automatically search for and compare space-time features.

6 Acknowledgments

The work was performed with the help of the members of the Laboratory for Visiometrics and Modeling at Rutgers University. The simulations were done on the CRAY-90 at the Pittsburgh Supercomputer Center and on the CM5 at NCSA. We gratefully acknowledge the support of the NASA Ames Research Center (NAG 2-829), DOE (DE-FG02-93ER25179.A000), ARPA (HPCD), and the CAIP Center, Rutgers University.

References

[1] D. Ballard and C. Brown. *Computer Vision.* Prentice-Hall, Inc, Englewood Cliffs, New Jersey, 1982.

[2] B. Jahne. *Digital Image Processing.* Springer Verlag, 1991.

[3] K. Ma, M. Cohen, and J. Painter. Volume Seeds: A Volume Exploration Technique. *The Journal of Visualization and Computer Animation*, 4(2):135–140, 1991.

[4] J. Miller, D. Breen, W. Lorensen, R. O'Bara, and M. Wozny. Geometrically Deformed Models: A Method for Extracting Closed Geometric Models from Volumes. In *Computer Graphics*, number 4, pages 217–226, July 1991.

[5] J. Udupa. 3D Visualization of Images. Technical Report MIPG196, Medical Image Processing Group, University of Pennsylvania, Philadelphia, PA, June 1993.

[6] D. Silver and N. Zabusky. Quantifying Visualizations for Reduced Modeling in Nonlinear Science: Extracting Structures from Data Sets. *Journal of Visual Communication and Image Representation*, 4(1):46–61, March 1993.

[7] I. Carlbom, I. Chakravarty, and W. Hsu. Integrating Computer Graphics, Computer Vision, and Image Processing in Scientific Applications. *Computer Graphics*, 26(1):8–16, January 1992. SIGGRAPH '91 Workshop Report.

[8] Y. Arnaud, M. Desbois, and J. Maizi. Automatic Tracking and Characterization of African Convective Systems on Meteosat Pictures. *American Meteorological Society*, pages 443–453, May 1992.

[9] J. Villasenor and A. Vincent. An Algorithm for Space Recognition and Time Tracking of Vorticity Tubes in Turbulence. *CVGIP: Image Understanding*, 55(1):27–35, January 1992.

[10] Y. Shinagawa and T. Kunii. Constructing a Reeb Graph Automatically from Cross Sections. *IEEE Computer Graphics and Application*, 11(6):45–51, November 1991.

[11] J. Helman and L. Hesselink. Representation and Display of Vector Field Topology in Fluid Flow Data Sets. *IEEE Computer*, August 1989.

[12] J. Wilhelms and A. Van Gelder. Octrees for Faster Isosurface Generation. *ACM Transactions on Graphics*, 11(3):201–227, July 1992.

[13] R. Samtaney. *Vorticity in Shock-Accelerated Density-Stratified Interfaces: An Analytical and Computational Study.* PhD thesis, Rutgers University, 1993.

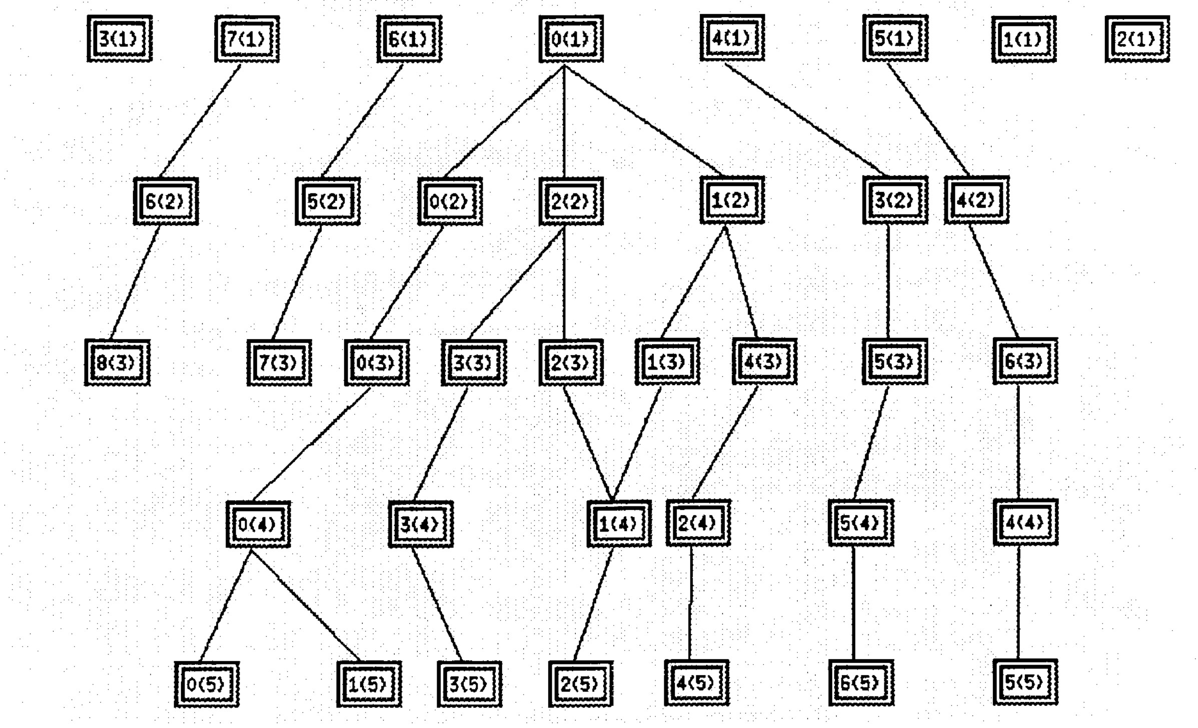

FIGURE 7. *History DAG for the 3D shock-interface simulation.*

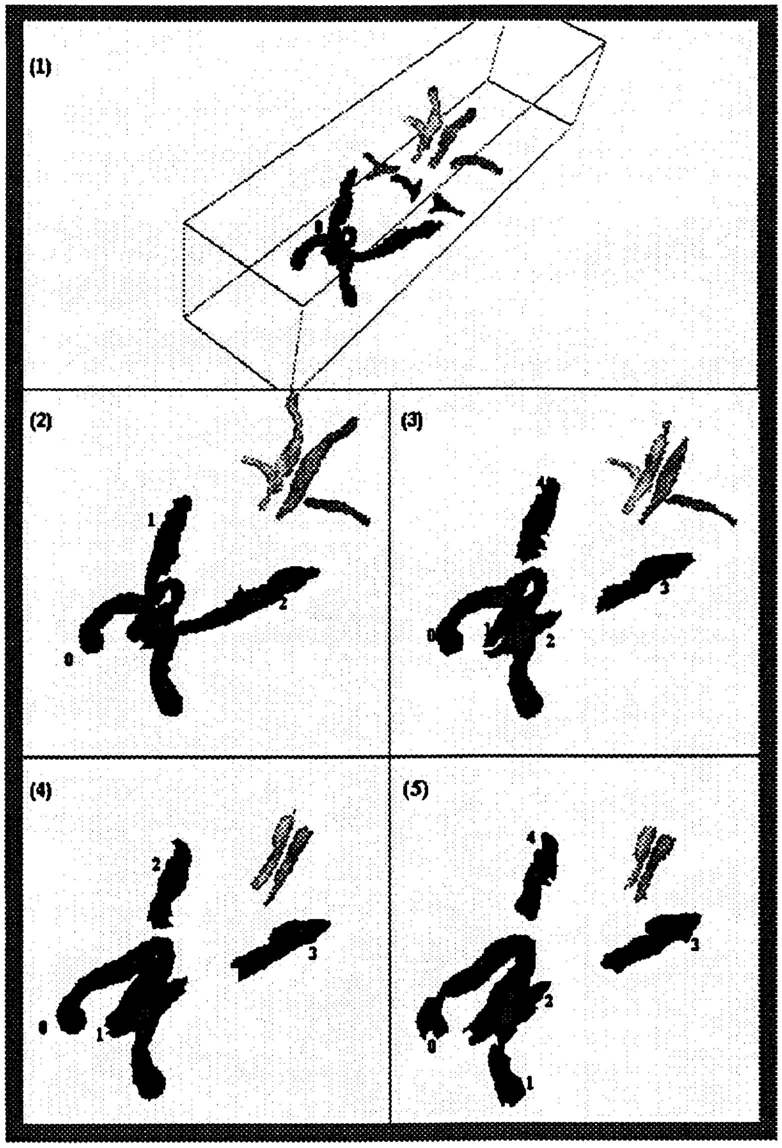

FIGURE 8. *Feature Tracking, 3D shock-interface, Air-Freon. Evolving regions are tracked and given the same color. Time = (1) 200, (2) 220, (3) 240, (4) 260, (5) 280*

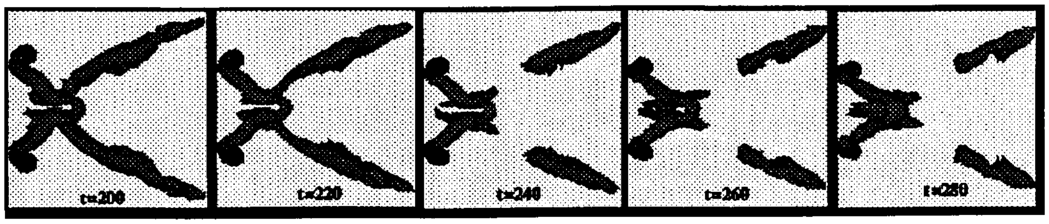

FIGURE 9. *The Hairpin Coherent Vortex Structure (CVS) is isolated and its evolution is tracked over time.*